**Spring boot学习总结**

1. mvn compile：

依赖注入

1. mvn package :

生成一个jar或者war,存放在项目的target中

1. mvn install :

打包并安装到本地仓库

1. mvn clean :

删除target中的包

5、@Service :

声明此类是业务处理类，通常与@Transactional一起配合使用

6、@Repository :

声明此类是数据库或者其他NoSQL访问类

7、@Component :

声明此类是一个Spring管理的类，通常用于无法用上述注解描述的Spring管 理类

8、@Configuration :

声明此类是一个配置类，通常与注解@Bean配合使用

@Configuration

public class DataSourceConfig {

@Bean(name = ” dataSource ”)

public DataSource datasource(Environment env) {

HikariDataSource ds =new HikariDataSource();

ds . setJdbcUrl(env . getProperty ( “spring .datasource .url” ) ) ;

ds.setUsername(env .getProperty (“spring .datasource.username”)); ds.setPassword(env.getProperty(“spring.datasource.password”) );

ds . setDriverClassName(env .getProperty (“spring.datasource .driver- class-name”));

return ds ;

}

}

9、@Bean :

作用在方法上，声明该方法执行的返回结果是一个Spring容器管理的Bean参考@Configuration示例

10、@PostConstruct注解和@preDestory注解 ：

当Bean被容器初始后，调用被@PostContruct注解修饰的方法

在容器被销毁之前，会调用被@PreDestory注解修饰的方法

11、spring boot 引入AOP ：

<dependency>

<groupid>org.springframework.boot</groupid>

<artifactid>spring-boot-starter-aop</artifactid>

</dependency>

12、@Aspect :

声明一个切面类

13、PUT：

类似于POST,PUT是更新操作，POST是新增操作

14、PATCH :

类似于PUT，表示局部更新操作

15、@RequestMapping中的属性：

consumes : 意味着请求HTTP头的Content-Type媒体类型与该属性值匹配才可以调用，例如：

@RequestMapping（path=”/test”,consumes=”application/json”）

produces : 与请求的HTTP头的Accept匹配上才可以调用该方法

16、 MultipartFile :

用于处理文件上传

例如：

@PostMapping(“/form”)

@ResponseBody

public FileBean uploadFile(String name, MultipartFile multipartFile){

}

如果上传多个文件用数组形式：MultipartFile[] files

MultipartFile 提供了以下方法来获取上传 的文件信息 :

(1)、getOriginalFilename: 获取上传的文件名字 ；

(2)、 getBytes：我取上传文件内容，转为字节数组；

(3)、 getlnputStream：获取一个 InputStream；

(4)、isEmpty：文件上传内容为空 ，或者就没有文件上传；

(5)、getSize： 文件上传的大小 ；

(6)、transferTo(File dest)： 保存上传文件到目标文件系统 。

文件上传的配置：

spring.servlet.multipart.enabled=true

spring.servlet.multipart.file-size-threshold=0

spring.servlet.multipart.location=

spring.serlet.multipart.max-file-size=lMB

spring.servlet.multipart.max-request-size=l0MB

spring.servlet.multipart.resolve-lazily=false

参数 enabled默认为true,即允许附件上传，file-size-threshold 限定了当上传的文件超过一定长度时，就先写到临时文件里。这有助于上传文件不占用过多的内存,单位是MB或者KB,默认是0，即不限定阔值。location指的是临时文件的存放目录，如果不设定，则是 Web服务器提供的一个临时目录。 max-file-size 属性指定了单个文件的最大长度，默认是lMB , max-request-size 属性说明单次HTTP 请求上传的最大长度，默认是 l0MB。 resolve-lazily 表示当文件和参数被访问的时候再解析成文件。

17、@RequestParam (name=” id”, required=true ) ：

属性： name指定参数的名称；

required为true时该参数必须赋值，false可以不传值；

defaultValue : 当参数没有赋值时会使用该默认值

18、@RequestBody :

客户端发起JSON请求，转化成对应的JavaBean

例如： @RequestBody User user;

19、@ModelAttribute :

注解 ModelAttribute 通常作用在 Controller 的某个方法上，此方法会首先被调用，并将方法结果作为Model的属性，然后再调用对应的Controller处理方法。

例如：

@ModelAttribute

public void findUserByid (@PathVariable Long id , Model model ) {

model .addAttribute (”user”, userService .getUserByid(id));

}

@GetMapping(path = ” /{ id}/get . json”)

@ResponseBody

public String getUser (Model model) {

System.out .println (model .containsAttri bute (”user ”) ) ;

return ” success”;

}

20、BindingResult ：

配合@Validated使用，参数的验证，存放验证结果

21、@AutoConfigureAfter（RedisAutoConfiguration.class） ：

表示此配置类需要在 RedisAutoConfiguration 配置类后再生效

22、@ConditionalOnBean：

在当前上下文中存在某个对象时，才会实例化一个 Bean

23、@ConditionalOnMissingBean：

在当前上下文中不存在某个对象时，才会实例化一个 Bean

1. @Scope:

默认是单例模式，即scope=singleton，另外还有prototype、request、session、global session作用域

(1)singleton: 全局有且仅有一个实例

(2)prototype: 每次获取Bean的时候都会有一个新的实例

(3)request: 表示针对每一次HTTP请求的时候都会产生一个新的Bean，同时该bean仅在当前HTTP request内有效

(4)session: 表示针对每一次HTTP请求的时候都会产生一个新的Bean，同时该bean仅在当前HTTP session内有效

(5)global session: